流程制作管理平台

二次开发说明文档

**［V1.0］**

**环球合一网络技术(北京)股份有限公司**

8B4BB0B3-793F-4E(03-31-15-26-07)

**文档信息**

|  |  |  |  |
| --- | --- | --- | --- |
| 项目名称: | 流程制作管理平台 | 项目编号： |  |
| 文档标题： | 《流程制作管理平台二次开发说明文档》 | 文档版本号： |  |
| 编写人: | 石得明 | 编写日期: |  |
| 审核人: |  | 审核日期: |  |

**文档版本履历**

| 版本号 | 修订日期 | 修订人 | 审核人 | 修订内容简述 |
| --- | --- | --- | --- | --- |
| V1.0 | 2018-02-06 | 石得明 |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目录

[目录 1](#_Toc505775775)

[1. 引言 2](#_Toc505775776)

[1.1. 编写目的 2](#_Toc505775777)

[1.2. 平台优点 2](#_Toc505775778)

[2. 系统架构 2](#_Toc505775779)

[3. 二次开发 3](#_Toc505775780)

[3.1. 支持任务类型 3](#_Toc505775781)

[3.1.1. 单次执行任务 3](#_Toc505775782)

[3.1.2. 循环执行任务 4](#_Toc505775783)

[3.1.3. quartz调度任务 4](#_Toc505775784)

[3.1.4. rest接口 4](#_Toc505775785)

[3.2. 各类型组件开发规范 4](#_Toc505775786)

[3.2.1. 单次执行任务 5](#_Toc505775787)

[3.2.2. 循环执行任务 5](#_Toc505775788)

[3.2.3. quartz调度任务 5](#_Toc505775789)

[3.2.4. rest接口 5](#_Toc505775790)

[3.3. 打包方式 5](#_Toc505775791)

[4. 解决方案及强调问题 5](#_Toc505775792)

[5. 附件 5](#_Toc505775793)

[5.1. pom.xml 5](#_Toc505775794)

[5.2. 组件配置文件模板 6](#_Toc505775795)

[5.3. 注解及异常控制流程包 6](#_Toc505775796)

1. 引言

## 编写目的

本文的主要对象是二次开发人员，通过研读本文，希望能让您进一步了解流程制作管理平台的功能，了解平台支持的任务类型，在此基础上，希望能在分解任务方面给您提供启发，指导您分解业务功能、划分组件功能范围。

## 平台优点

* 无单点。平台、容器、zookeeper均可以集群部署
* 组件开发简单，降低开发人员门槛
* 支持多种组件类型，目前支持的有：单次运行任务、周期运行任务、quartz调度任务、rest接口发布（可以转通道）。解决了传统服务管理系统只能管理http接口的问题
* 组件统一管理、执行统一管理、容器统一管理
* 组件产生的中间数据存放在rabbitmq上，rabbitmq支持集群部署，仍不存在单点。消息队列中间件方便替换，平台添加新的消息中间件，组件完全不需要修改，只改流程的配置即可
* 流程执行中间出现异常退出或者宕机，数据不会丢失

## 统一名词

|  |  |
| --- | --- |
| 简称 | 解释 |
| 平台 | 流程制作管理平台的管理页面程序 |
| 容器 | 流程制作管理平台执行流程配置程序 |
| 组件包 | 一个或者多个组件按照规范打的JAR包或者ZIP包 |
| 组件 | 一个特定的功能 |
| 流程 | 完成特定业务功能的一个或者多个组件的集合，一般情况下，相当于一个生产流水线 |
| 流程执行配置 | 容器可以执行的流程数据，与流程不同之处是：流程的主要数据都是json格式；流程执行配置，把流程的json数据解析为对象，方便容器使用。流程执行配置分平台中的流程执行配置和容器中的流程执行配置，平台中的流程执行配置包含有流程的对象，在原来流程已经删除的情况下，可以由流程执行配置恢复以前流程，方便再次编辑。为了减小容器的内存使用，在流程执行配置添加到容器上时，删除了跟执行没关的对象。执行配置一旦添加到容器上执行后，其属性不能再被修改，如果需要修改属性，需要在平台上原有的流程执行配置上修改，然后删除容器中的流程执行配置，重新添加。 |
| comID | 一个或者一组组件的唯一标识，同一个comID的组件必须同时注册，同时卸载。不同的组件包不能有相同的comID。 |
| 组 | 同一个业务方向的组件提供归类的依据，主要用于画流程图时方便展示 |
| 组件类 | 组件实现的类class，有时跟组件混用 |
| 组件方法 | 组件实现功能的入口方法，系统会自动调用此方法 |
| 初始方法 | 系统在调用组件方法前调用此方法，用来初始化功能资源 |
| 关闭方法 | 流程或者组件在关闭时调用的方法，用来回收、释放资源 |
| 执行类型 | 任务类型 |
| 组件名称 | 组件的英文名称，一般为组件类的简写 |
| 组件中文名 | 组件的中文意名，主要在画流程图时分辨其含义 |
| 需要读 | 即组件执行时，需要从MQ中读取数据。对应的组件执行方法必须有并且只能有一个Object类型的参数 |
| 需要写 | 即组件执行结束时，可以返回一个结果，画流程图时，如果该组件后面跟一个组件，系统会把此返回值写入配置的MQ队列中。对应的组件执行方法必须有返回值。画流程时该组件后面可以不用跟组件，此时系统会把返回值丢弃。 |
| 通道信息 | 组件间交互数据的MQ队列 |
| 开发属性配置 | 系统框架需要的属性，这些属性可以配置到组件类中。系统运行时，判断出配置了开发属性配置后，把该属性值赋给框架相应的对象。 |
| 组件属性配置 | 组件需要的属性 |

1. 系统架构

流程制作管理平台分为平台管理系统和流程执行容器系统，以下称平台和容器，各模式结构如下：

 平台之间通过zookeeper同步数据，做到各平台之间的数据保持一致。平台和容器之间通过RPC服务交互数据，RPC服务是由流程制作管理平台提供的基于zookeeper实现的RPC服务。

1. 二次开发

## 支持任务类型

二次开发前，先需要了解一下目前系统支持哪些类型的组件，及其功能。目前支持的任务类型有：单次执行的任务、周期执行的任务、quartz调度任务、rest接口（支持转通道）。下面依次介绍：

### 单次执行任务

顾名思义就是一次启动只执行一次。可以多次启动，但每次启动只执行一次。

此种类型的组件适合于不定时、随时有可能需要执行的功能，例如编辑需要导出片单的工作。

### 循环执行任务

此种任务是周期性执行任务，多用于处理从MQ队列中取数据，经过处理把结果再写回下一个MQ队列或者写入DB，可以配置循环周期控制处理速度。

### quartz调度任务

此种任务集成了开源框架quartz功能，把组件的功能包装一下交由quartz调度执行。与循环执行任务不同，本任务不能与MQ中间件交互，不能从MQ队列中读数据。此任务继承了quartz调度的灵活性，比循环执行任务能更精确地控制任务的执行时间。多用于每周导出注入片单或者统计注入数据。

### rest接口

提供HTTP接口服务。本功能主要用于与其它系统交互的功能，在业务流程开始时，提供数据入口，其它系统调用HTTP类型的任务接口，容器把业务数据转到MQ上，再异步处理。此功能在其它系统没有实现MQ客户端功能时，提供HTTP接口，把数据转到MQ上，减轻了与其它系统对接难度。

## 各类型组件开发规范

组件开发前，首先您需要先对业务进行功能拆分，分解成一个个尽可能小的功能组件，以方便流程灵活组合、增减组件；其次，需要对功能进行分析，不同的功能组件，选择合适的任务类型、数据来源，例如，需要重试或者需要等待的任务，数据来源尽量是数据库，按状态值周期读取处理；跟其它系统对接时，可以首先提供一个REST接口组件，把其它系统发过来的数据，先存到MQ中间件中，异步处理；最后根据确定下来的功能组件，设计合适的数据类型用于各组件间交互数据，并设计出合适的表结构存储这些任务。至此，就可以按照下面的规范开发组件。

### 开发原则和注意事项

1. 组件类名全局唯一，不同的comID下的组件类名也不能重复。
2. 同一个组件包里尽量只有一个组件，一个组件修改、重新注册不影响其它组件。如果多个组件打到同一个组件包里，这些组件的comID必须一样。
3. 平台中各组件的jar包是隔离的，因此各组件打进的依赖包可以有不同版本，不会出现问题。同时，如果各组件间用到同样的实体类，则需要每个组件包都要打进这些实体，否则没打进去实体类的组件，将报找不到类异常。也可以把实体类单独打bean包，放到平台和容器的ext文件夹下，重新启动平台和容器。由此，同一容器中运行的各组件，如果不是由同一个组件包中注册进平台的，使用同一个单例模式的对象时将不再生效
4. 如果使用配置文件描述组件信息，配置文件必须以com.xml结尾。
5. 如果组件中使用了spring容器，spring的配置文件需要以 comID+spring.xml结尾。
6. 二次开发过程中用到的额外的jar包，都放在ext目录下，尽量不要增减lib下面的jar包。在向ext下添加包时，确认lib下面是否已经存在或者存在不同版本的jar包。
7. 使用注解标记组件类时，需要在二次开发工程里导入附件中的《注解及异常控制流程包》。
8. 如果组件需要从通道里读数据，开发组件时，只要在执行方法上添加一个Object类型的参数，该参数具体是什么类型，需要开发组件前预先知道从哪个组件里收数据，该参数值即是上个组件返回的值的类型。注意：参数的类型必须是Object，处理时把对象强制转换成真实类型。只能有一个参数。总结如下：如果组件需要从通道里取数据处理，则执行方法有且只能有一个Object类型的参数，如果不需要从通道里处理数据，则不能有参数；如果组件需要往通道里写数据，则执行方法的返回类型不能是void，可以不是Object。
9. 如果一个组件需要返回多个值，分别写到通道里，供下个组件处理，只需要把返回值添加到List里面，返回值类型是List即可。
10. rest接口类型的组件，如果需要把结果写到通道里，只能在rest方法上添加Writeable注解，暂时不支持用配置文件描述该功能。
11. rest接口的地址必须以rest开头。
12. 不管配置文件或者注解上写的comID是大写或者小写，进入系统后，全部是小写字母。
13. 如果用注解的方式描述rest接口的组件，必须用 @RestController和@ComponentAnn同时使用

### 单次执行任务

采用注解方式的实现，直接上例子：

**package** com.guttv.com.sample.once.ann;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

**import** com.guttv.pm.support.ann.CloseMethod;

**import** com.guttv.pm.support.ann.ComponentAnn;

**import** com.guttv.pm.support.ann.ExecuteMethod;

**import** com.guttv.pm.support.ann.InitMethod;

**import** com.guttv.pm.support.ann.ProAnn;

**import** com.guttv.pm.support.ann.ExecuteMethod.ExecuteType;

@ComponentAnn(cn = "执行一次样例", group = "sample", comID = "onceAnnSample")

**public** **class** OnceComponent {

**protected** Logger logger = LoggerFactory.*getLogger*(**this**.getClass());

@ExecuteMethod(type = ExecuteType.***Once***)

**public** **void** once() {

logger.info("一次执行只能看到一次");

}

//该方法可有可无

@InitMethod

**public** **void** init() {

}

@CloseMethod

**public** **void** close() {

}

@ProAnn(cn="中文名称")

**private** String param = "default value";

}

组件包： 

采用配置文件的实现，直接上例子：

package com.guttv.com.sample.once.config;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

public class OnceComponent {

protected Logger logger = LoggerFactory.getLogger(this.getClass());

public void once() {

logger.info("一次执行只能看到一次");

}

public void init() {

}

public void close() {

}

}

组件包： 

### 循环执行任务

组件可以与spring结合，框架从spring容器中获取组件对象，对象中需要注入的对象，也可以按spring的注入规则注入进去。其它类型的任务均可以按spring的规则注入。目前平台内各种类型的任务如果想注入其它springbean都需要提供一个spring的配置文件。

上组件包：

### quartz调度任务

详见源码

### rest接口

请求前缀必须是rest，同一种匹配路径可以注册多个组件，但同时只能启动一个提供服务。

详见源码

### 样例原码



## 打包方式

目前系统支持两种格式的文件注册组件：zip和jar。

pom文件里有的JAR包不用打进包，如果不在pom文件里，少量的JAR包，可以把组件JAR包和依赖包放在一起打成ZIP包，如果大量的JAR包，可以把依赖的JAR包放在PM\_HOME/ext文件夹下，重启工程。理论上组件类如果不直接依赖的JAR包不用把JAR包放在平台的 ext文件夹下，只需要放在容器的ext目录下即可。附件中的 注解及异常控制流程包不用打进ZIP包中。

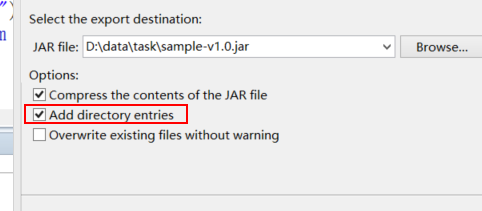
如果组件包是ZIP，组件的配置描述文件只能打在zip目录里，不能打在zip包中的jar包里；组件使用的spring配置文件必须打在zip包中的jar包中。如果组件包是JAR，组件的配置描述文件可以打在jar中的任何目录，spring配置文件也可以打在jar包的任何位置。

如果用到数据库配置信息，建议把jdbc.properties文件放在 config下面，不要打在zip或者jar包中，在spring配置里可以直接引用classpath:jdbc.properties

业务用到的其它不经常改变的配置文件也可以放在 pm/config下面，通过ClassUtils.getDefaultClassLoader().getResource("xxx")获取文件。

1. 常见问题及解决办法
2. 问题：写好组件，打好包注册时报没有找到注解的组件

原因：打包时的问题，打包时选择：add directory entries



1. 附件

## pom.xml



## 组件配置文件模板

强调：如果使用配置文件的方式注册组件，配置文件必须以com.xml结尾。



## 注解及流程控制异常包

